In the Claims:

1. (currently amended): A visual linker, comprising:
an allocation module for allocating sections of code and data into different target memories ~~memory~~ of a processor without running a confidence check;
an output module for writing a file of said allocated sections of code and data;
an incomplete link module, wherein said incomplete link comprises allocation information on those sections that are allocated by said allocation module and those that have not yet been allocated without running a confidence check and without actually completing the link; said allocation information including the allocated position and size of those sections that are allocated to said different target memories;
a link server that interprets linking instructions, modifies said incomplete link module accordingly and invokes said allocation module accordingly and invokes said output module accordingly;
an interface for receiving instructions for said link server and for providing feedback as to the state of said incomplete link;
and a graphical user interface that generates said instructions in response to user gestures and graphically displays the state of said incomplete link.

2. (currently amended): A method of incrementally and interactively allocating code and data sections, comprising the steps of:
generate a specific allocation instruction from a client program or program component;
executing said instruction by making alterations to allocation information associated with one or more code or data sections;

resolving allocation to the full extent possible given current allocation information associated with all code and data sections involved in a link;
recording linking instructions received that describe how the visual linker is to be controlled and so that said linking instructions can be replayed without interaction to obtain the same linking effect;
report to client programs current allocation state inclusive of allocation errors and sections not yet allocated;
and repeating these steps until all sections of code and data have been allocated.

3. (original): The method of Claim 2 including the step of interpreting a user gesture made to a graphical user interface as a specific allocation instruction.
4. (original): The method of Claim 3 wherein a gesture is a drag-and-drop operation or a point-and-click operation on a video screen.
5. (original):The method of Claim 2 including the step of displaying said current allocation state graphically to the user, inclusive of allocation errors and sections not yet allocated.
6. (original):The method of Claim 2 including the step of writing the results to an output file.
7. (canceled)
8. (currently amended):The method of Claim ~~7~~ 2 wherein the record of link instructions may be displayed and altered through a graphical user interface.
9. (original):The method of Claim 2 wherein the set of code and data sections involved in the link may be determined by a reachability check performed using a cross-reference graph.
10. (original):The method of Claim 2 wherein an instruction may specify an overflow policy to be used

whenever a portion of memory fills up during allocation.

11. (original):The method of Claim 2 wherein an instruction may provide for an allocator-optimized stack or heap size by specifying a minimum and maximum size instead of a particular size.
12. (original):The method of Claim 2 wherein one instruction may apply allocation operations to a related group of sections simultaneously.
13. (original):The method of Claim 12 wherein the sections may be the set of sections reachable from a specific starting section, as determined by a cross-reference graph.
14. (new) The method of Claim 2 including the step of replaying said linking instructions without interaction to obtain the same linking effect.
15. (new) The method of claim 14 including the steps of displaying the record of the link instructions and altering through a graphical user interface.
16. (new): The linker of claim 1 wherein said link server and a link recipe store linking instructions received that describe how the visual linker is to be controlled and can be replayed without interaction to obtain the same effect as the sequence of commands or gestures.
17. (new): The linker of Claim 16 wherein said stored linking instruction are displayed and altered through said graphical user interface.
18. (new): A visual linker, comprising:
an allocation module for allocating sections of code and data into different target memories of a processor including fast on-chip memory;
an output module for writing a file of said allocated sections of code and data;
an incomplete link module, wherein said incomplete link comprises allocation information on those

sections that are allocated by said allocation module and those that have not yet been allocated without actually completing the link; said allocation information including the allocated position and size of those sections that are allocated to said different target memories including said fast on-chip memory;

a link server that interprets linking instructions, modifies said incomplete link module accordingly and invokes said allocation module accordingly and invokes said output module accordingly;

an interface for receiving instructions for said link server and for providing feedback as to the state of said incomplete link;

and a graphical user interface that generates said instructions in response to user gestures and graphically displays the state of said incomplete link.

19. (new) The linker of claim 18 wherein said link server and a link recipe store linking instructions received that describe how the visual linker is to be controlled and can be replayed without interaction to obtain the same effect as the sequence of commands or gestures.

20. (new): The linker of Claim 19 wherein said stored linking instruction are displayed and altered through said graphical user interface.